ORACLE® ACADEMY

Java Fundamentals

3-1
Getting Started with Greenfoot

Objectives

This lesson covers the following objectives:

- Download and install Greenfoot
- Describe the components of the Greenfoot interactive development environment
- Create an instance of a class
- Describe classes and subclasses
- Recognize Java syntax used to correctly create a subclass

Launch Greenfoot

To launch Greenfoot:

- Double-click the Greenfoot icon on your desktop.
- Select the Greenfoot program from your list of computer programs.

Greenfoot Textbook Scenarios

- To become familiar with Greenfoot, download and run the scenarios created by the authors of the Greenfoot textbook.

A scenario is a game or simulation implemented in Greenfoot.

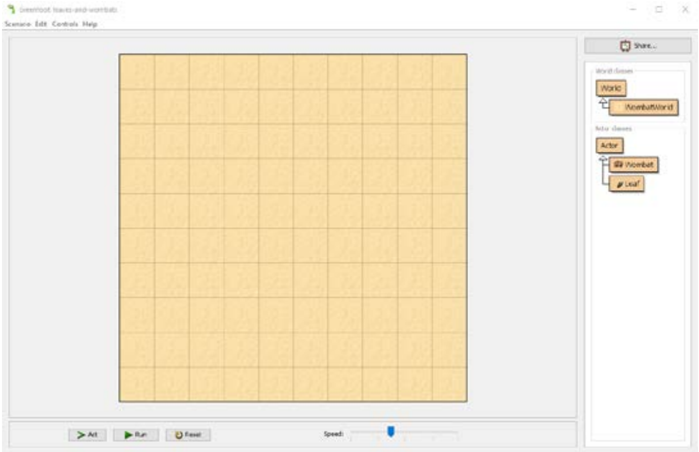
- Instructions for working with existing scenarios are given on the following slides.

Steps to Download the Greenfoot Textbook Scenarios

- Go to the Greenfoot textbook webpage:
  - http://www.greenfoot.org/book
- Click the Book Scenarios link.
- Save the zip file to a folder on your computer.
- Extract the zip files to a folder on your computer.
- Name this folder "Greenfoot Scenarios".
- If your computer does not have zip file extraction software, download free, open source software at 7zip.com.

Steps to Open a Scenario in Greenfoot

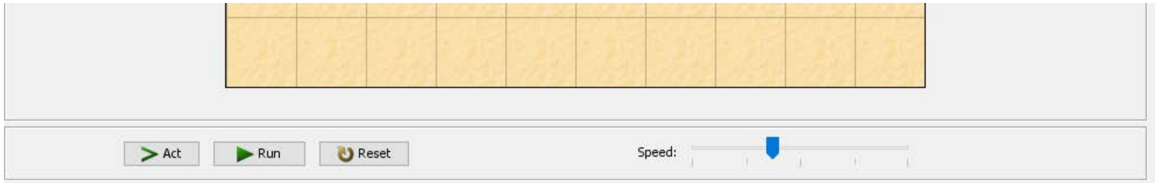
- From the Scenario menu, select Open.
- From the Greenfoot scenarios folder you created on your computer, select the leaves-and-wombats scenario from the chapter01 folder. (Answer “Yes”, if asked to update code)
- The scenario will open in a new window.



Execution Controls

Execution controls to run a scenario include:

- Act: Runs all actions in the scenario once.
- Run/Pause: Runs all actions in the scenario repeatedly until Pause is clicked.
- Reset: Pauses the scenario or resets the scenario back to its starting position.
- Speed: Runs actions faster or slower.



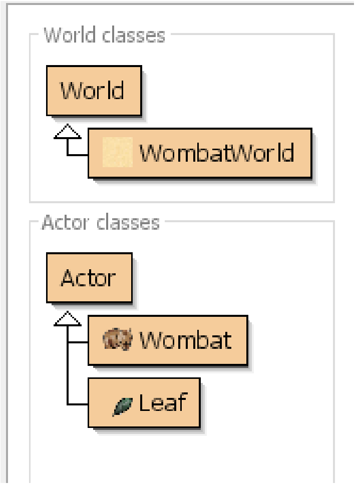
Defining Class Characteristics

- In nature, a bee has characteristics that are common to all bees: six legs and two wings.
- The bee may also inherit the characteristics of its breed that gives it a specific color, shape and size.
- In Greenfoot we would create a Bee class that defines these characteristics. This defines how all bee objects included in the scenario look and act.

> A class contains the specifications that define the appearance and movement of an object. The class provides instructions to Greenfoot for how to create and display instances when they are added to your scenario.

Classes in Greenfoot

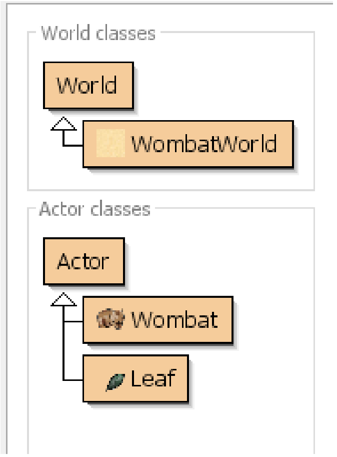
- The class tells your scenario how its objects should look and act when the scenario is run.
- When you add a class to your scenario, it appears in the class hierarchy (to the right of the world).
- You can add as many instances of the class as you wish to the scenario.



Superclass Types

Two types of superclasses appear in the Greenfoot class hierarchy:

- World:
  - Holds the subclasses that provide the background image for the scenario's world
  - Defines the size and resolution of the world
- Actor:
  - Holds the subclasses that produce the instances that act in the scenario.



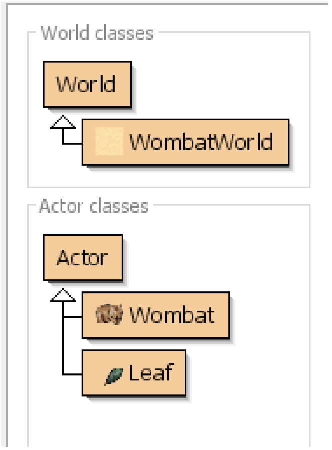
The overarching class of a group of classes is called a superclass. In this example - World and Actor

Subclasses

- Subclasses are a specialization of a class.
- For example, the Wombat class is a subclass of the Actor superclass. This subclass:
  - Inherits all of the properties of the Actor superclass, such as a pre-defined set of actions that Actor subclasses can perform.
  - Has properties specific to its subclass, such as the image that gives Wombat objects their appearance.
  - Can receive new properties that the programmer creates specifically for the subclass, such as images or actions.

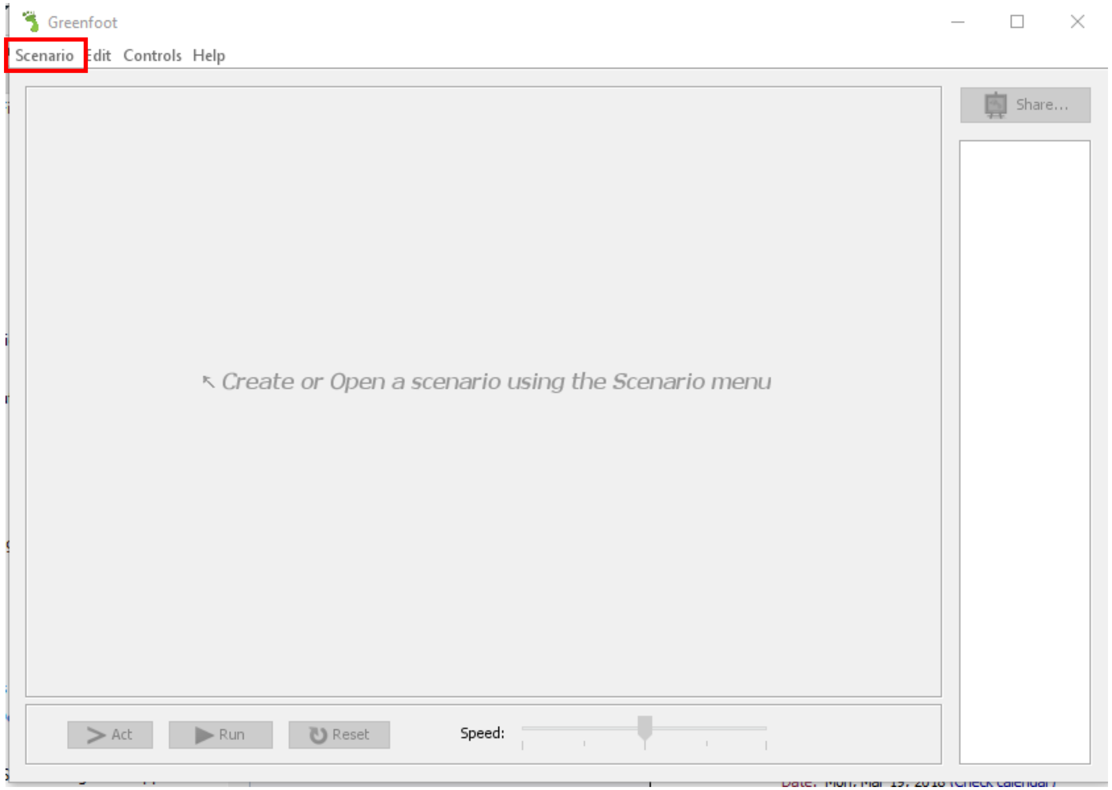
Subclass Properties

- A subclass has an "is-a" relationship to a superclass (Wombat is a subclass of the Actor superclass).
- Properties can be modified (such as the class's name, image to display, or actions to perform).
- An arrow in the class hierarchy shows the subclass's relationship to the superclass.



Try It – Create a new scenario

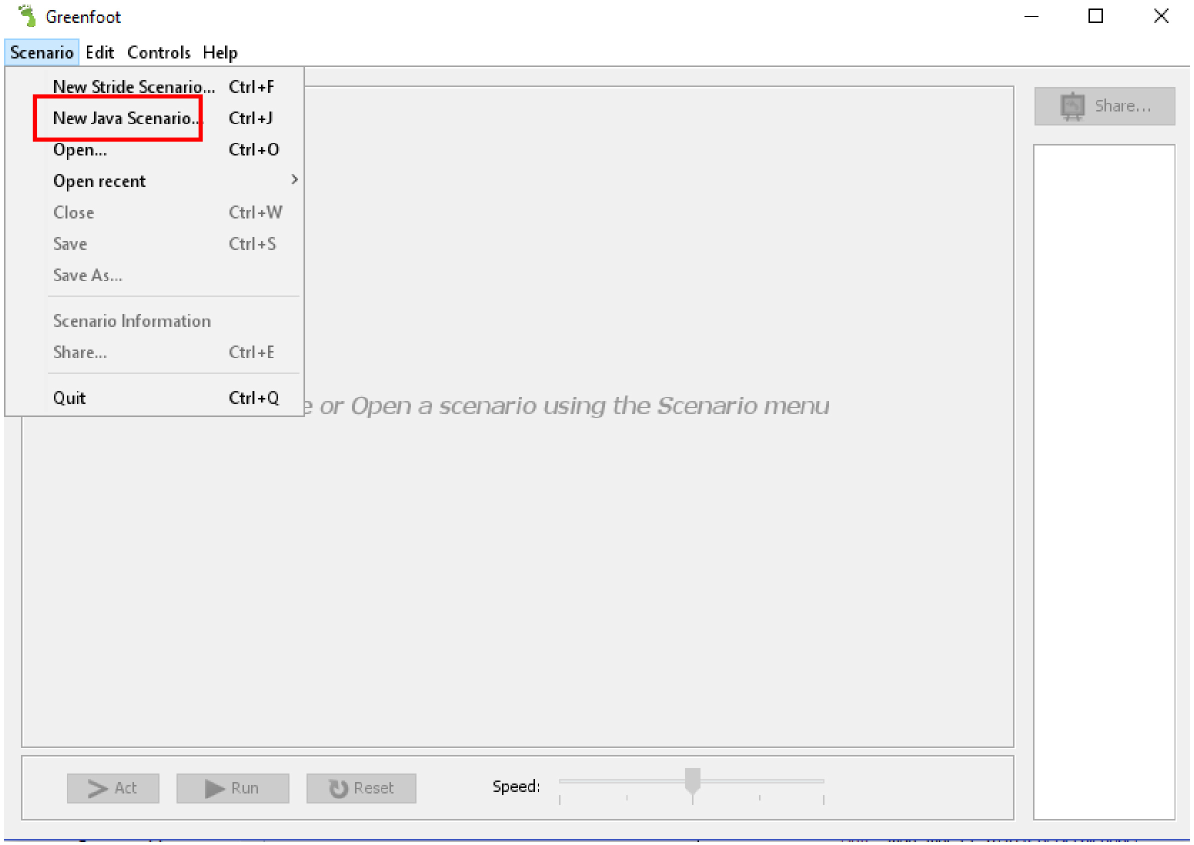
- Open Greenfoot
- Select the Scenario Menu



↖ Create or Open a scenario using the Scenario menu

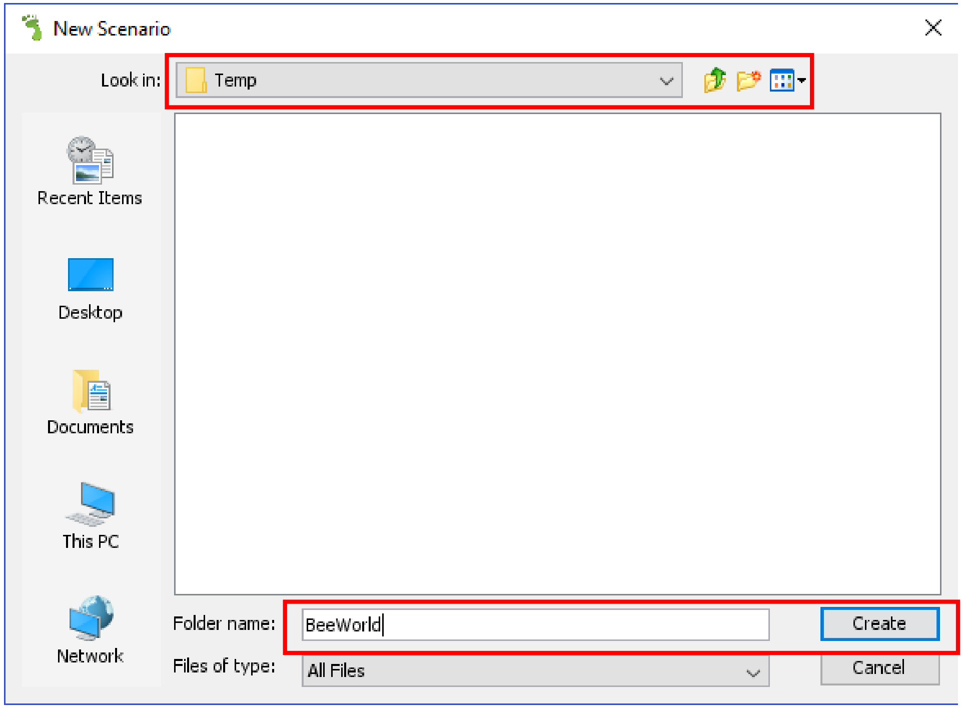
Try It – Create a new scenario

- Select New Java Scenario...



Try It – Create a new scenario

- Navigate to where you plan to save your Greenfoot files, and for “Folder name:” type “BeeWorld”
- Click “Create”



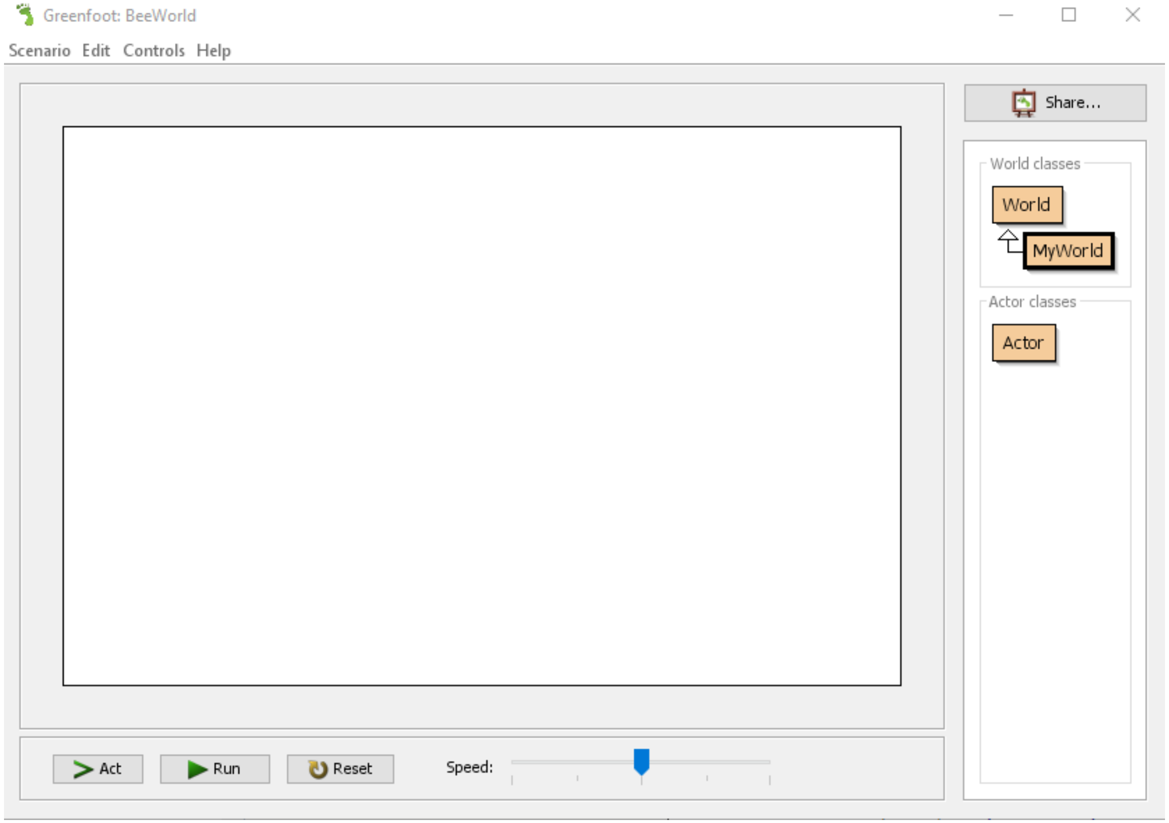
Look in: Temp

Folder name: BeeWorld

Files of type: All Files

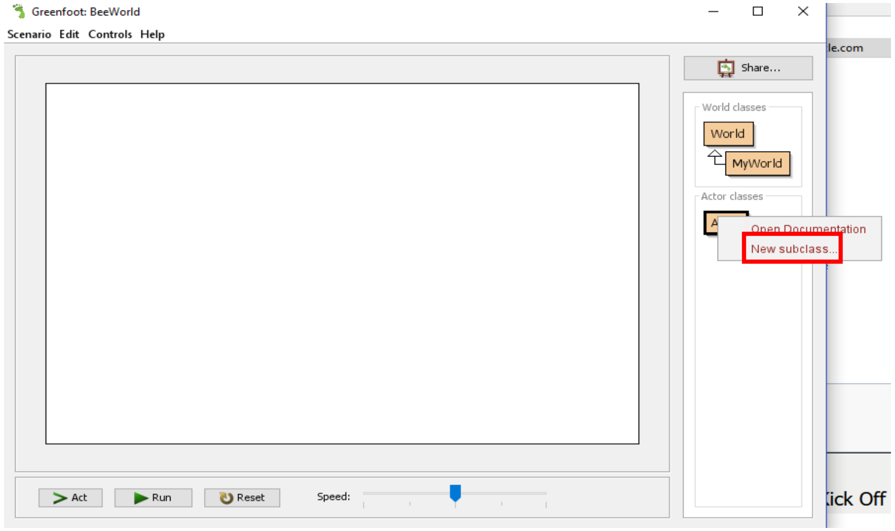
Try it – Create a new scenario

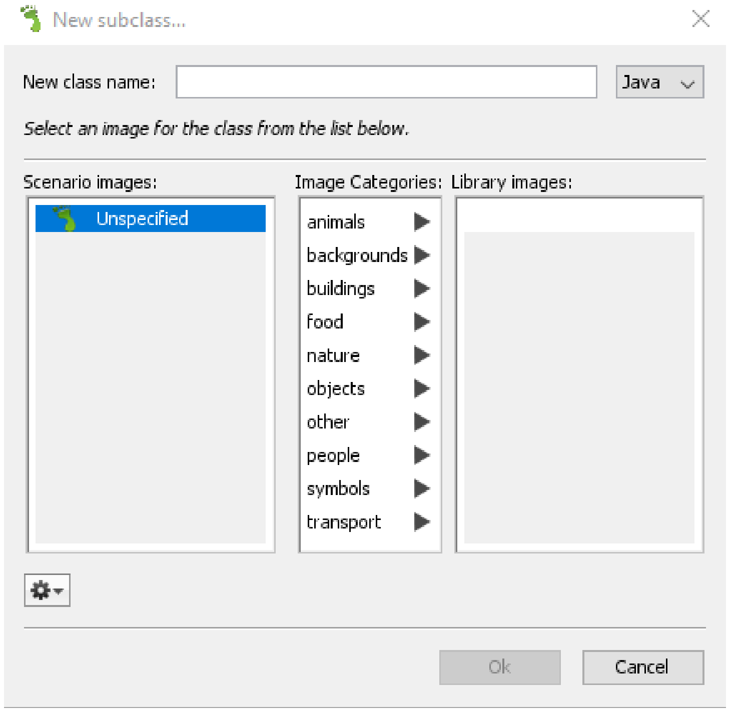
Your scenario should look like this:



Try it – Create a new scenario

- Next, you will add a Spider, a Bee, and a Fly
- Right+click on the Actor class, and select New subclass

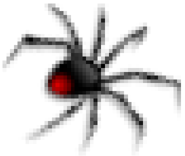


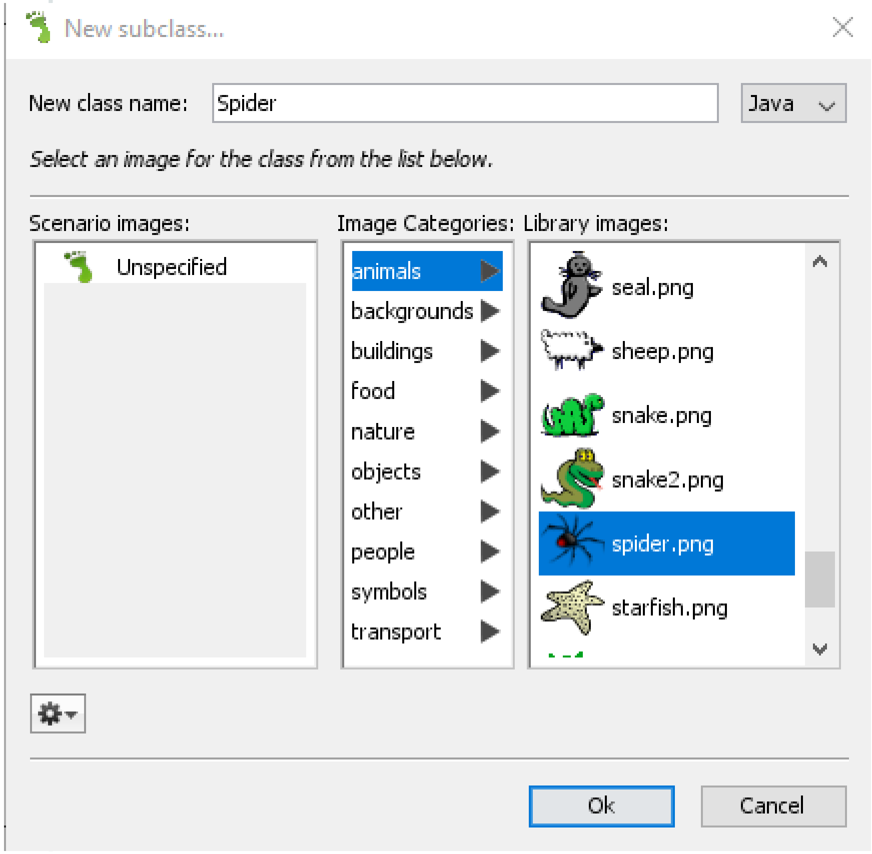


Select an image for the class from the list below.

Try it – Create a new scenario

- Type “Spider” as the New class name:
- Select the “animals” Image Category
- Select “spider.png” from Library Images list
- Click “Ok”



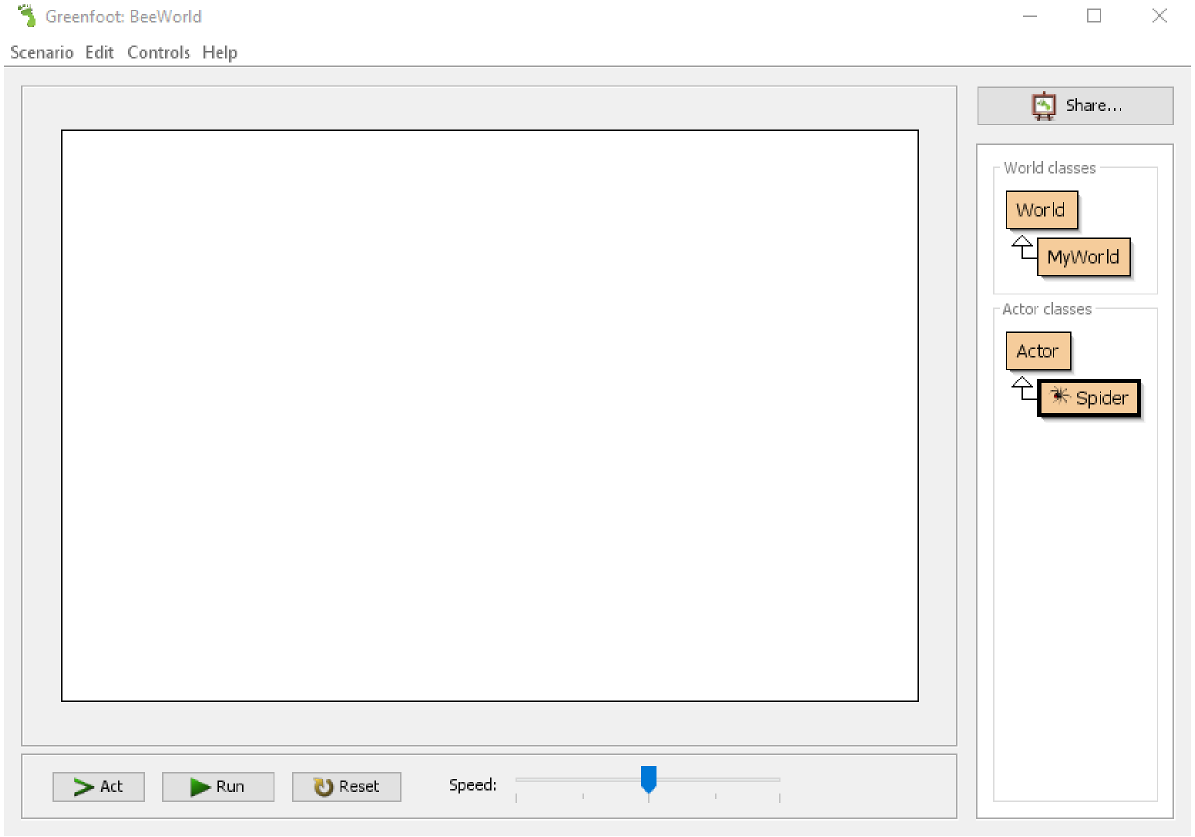


New class name: Spider

Select an image for the class from the list below.

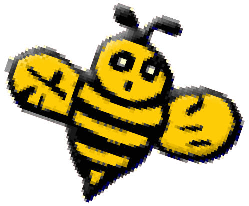
Try it – Create a new scenario

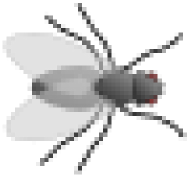
Your results should look like this:

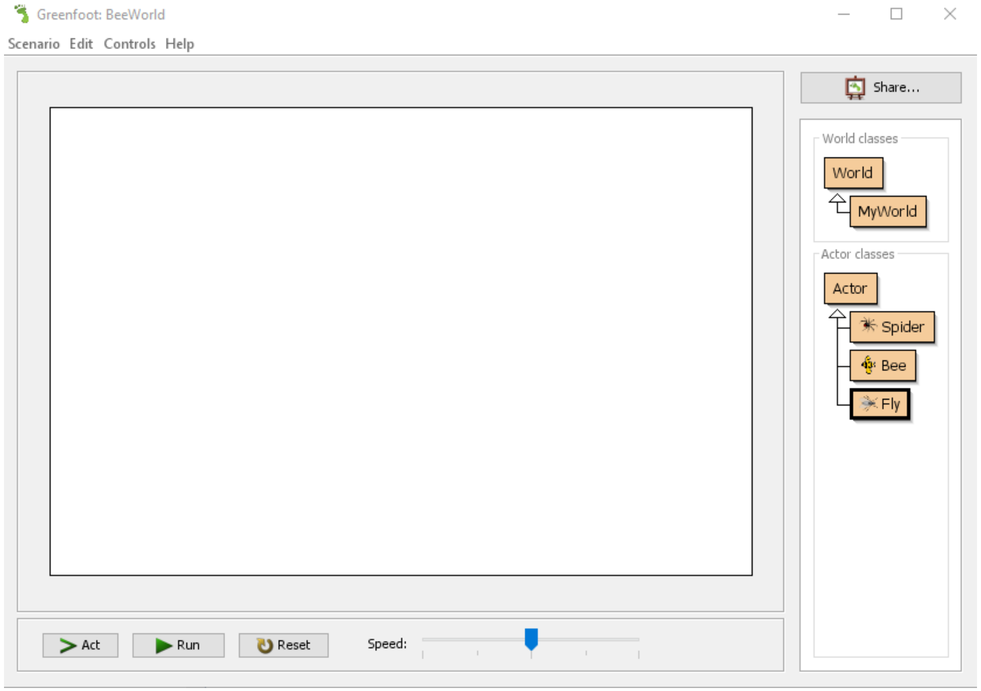


Try it – Create a new scenario

- Now add a Bee and a Fly
- Your results should look like this:

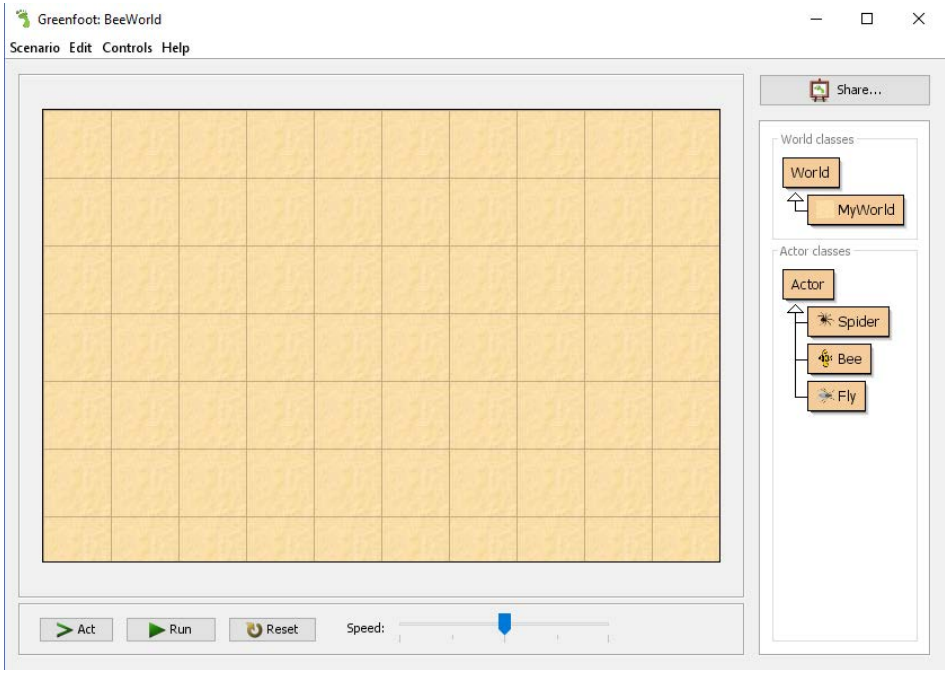






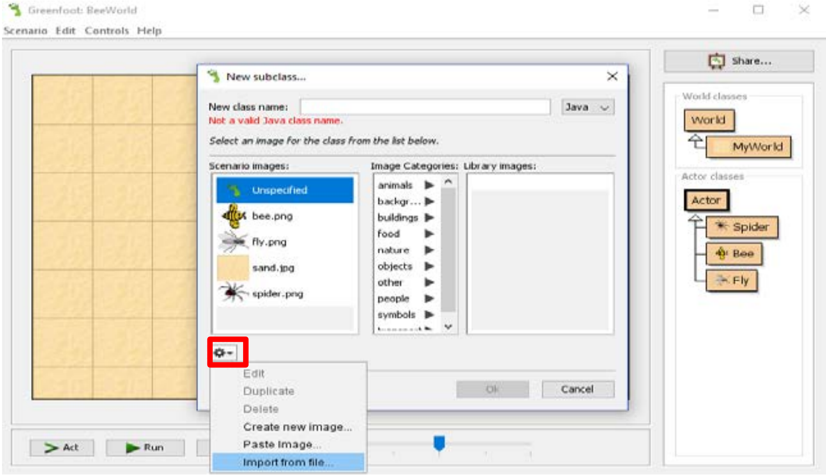
Try it – Create a new scenario

- Final step – Add a background to your scenario
- Right+click on MyWorld, and choose "Set Image"
- From the "backgrounds" Image Categories, select sand.jpg
- Click "Ok"



Steps to Create a New Subclass with an Image Imported From Your Computer

- Right click on the World or Actor superclass.
- Select New Subclass...
- In the New Class window, select the “Gear” icon, and choose Import From File...
- Navigate to an image on your computer, name it, then click “Ok”


New subclass...

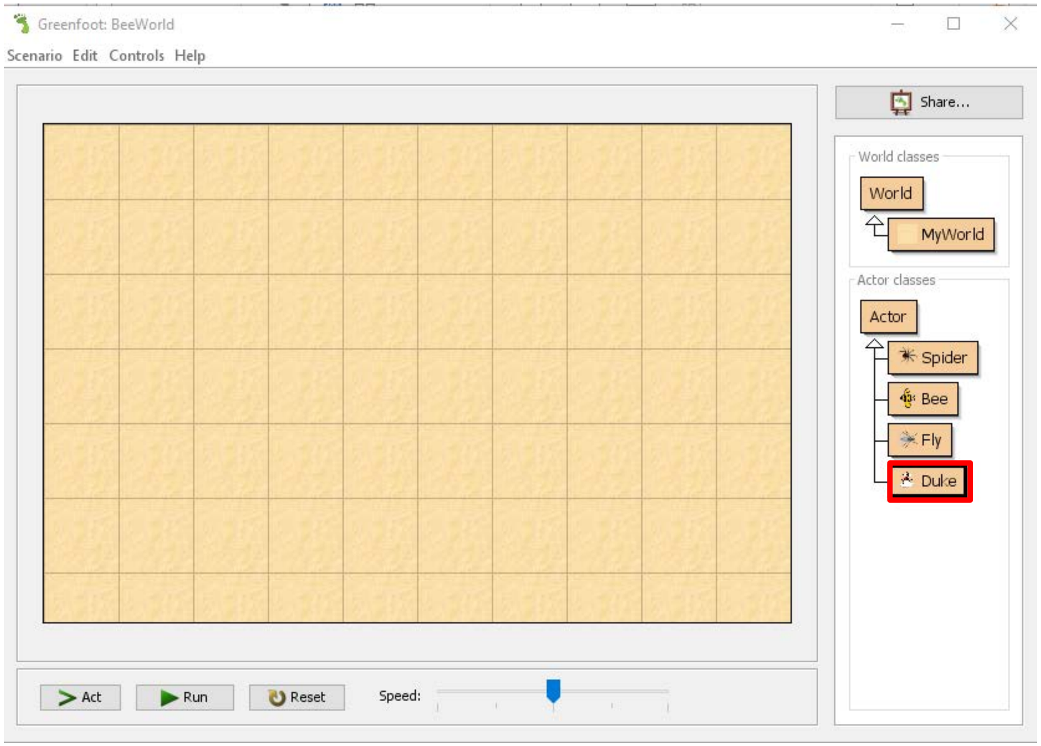
New class name:

Not a valid Java class name.

Select an image for the class from the list below.

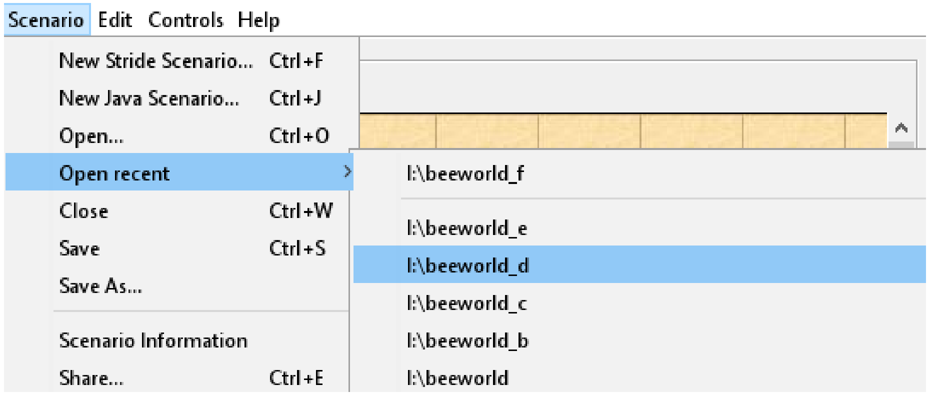
Steps to Create a New Subclass with an Image Imported From Your Computer

In this example, we have imported an image of the Java Icon, called “Duke”



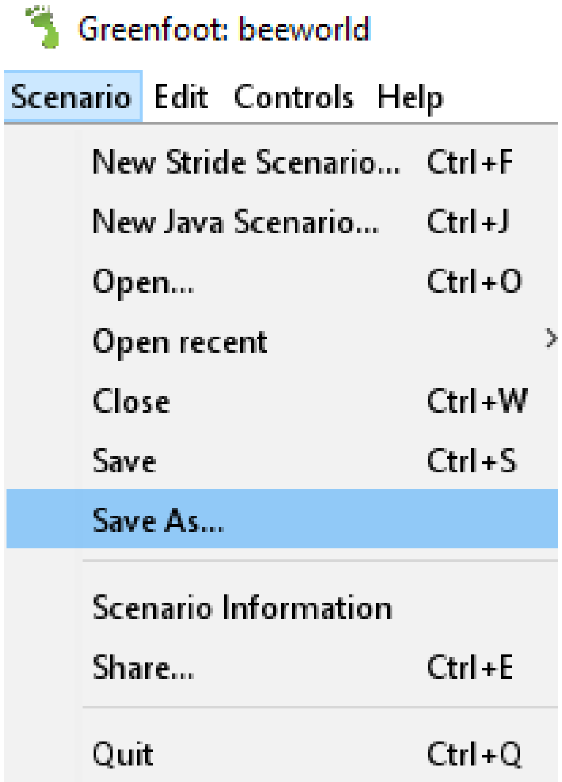
Save Multiple Versions of Scenarios

- Save the scenario frequently as you work.
- Each time you close Greenfoot, it saves your current work.
- Save multiple versions of scenarios:
  - To return to an earlier version of a scenario.
  - To have multiple scenarios to work from.

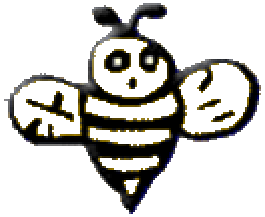


Steps to Save a Scenario

- In the Scenario menu, select Save As...
- Save a copy to a folder on your computer.



Instances of a Class



- The Bee class defines the characteristics of a Bee. Such as: movement, color, size etc.

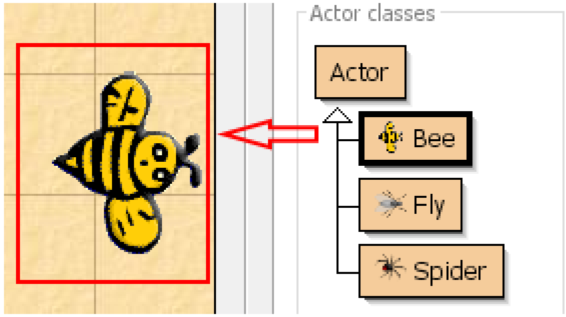
- A Bee that flies in a field, or rests on a flower, is a physical object that is a unique instance of the Bee class.

- An instance object holds its own unique set of characteristics as defined in the class, but can be manipulated and changed.

Greenfoot Instances

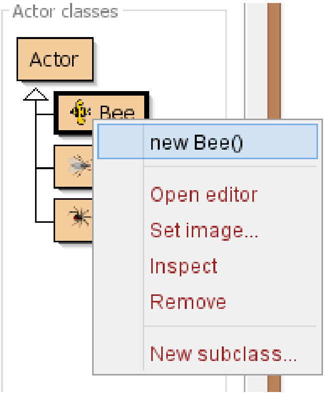
- One or many instances of a class can be added to the scenario.
  - Actor instances move and act in your scenario.
  - World instances provide the background for your scenario.
- Instances can perform the behaviors written by the programmer in the class's source code.

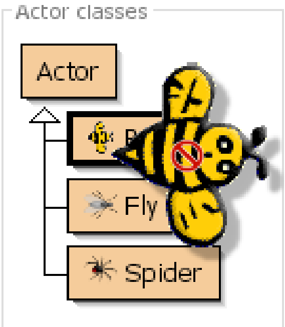
Instances are the objects from a class that act in your scenario.

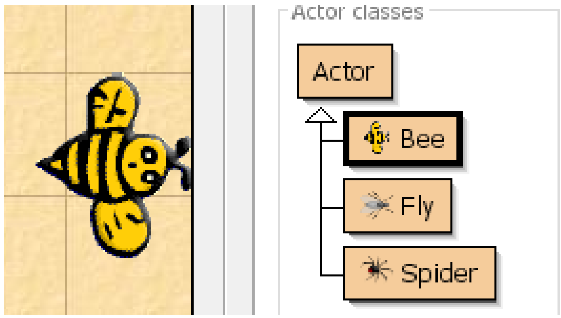


Steps to Add an Instance to a Scenario

- Right click on the class.
- Click the new [class name] option.
- Drag the instance into the scenario with your cursor.
- Later, you will program the instance to act by writing source code in the class's Code editor.

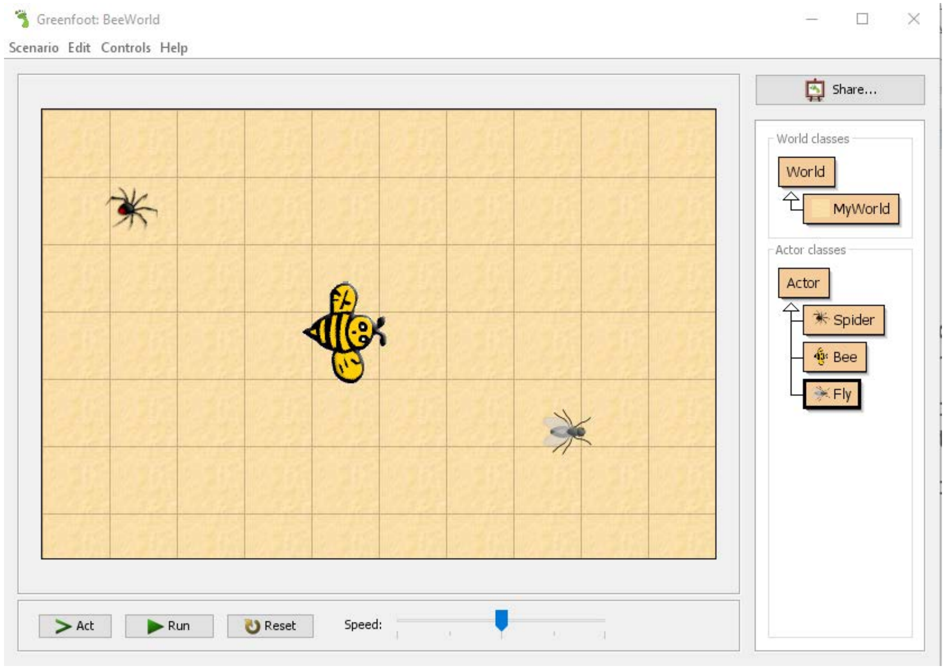






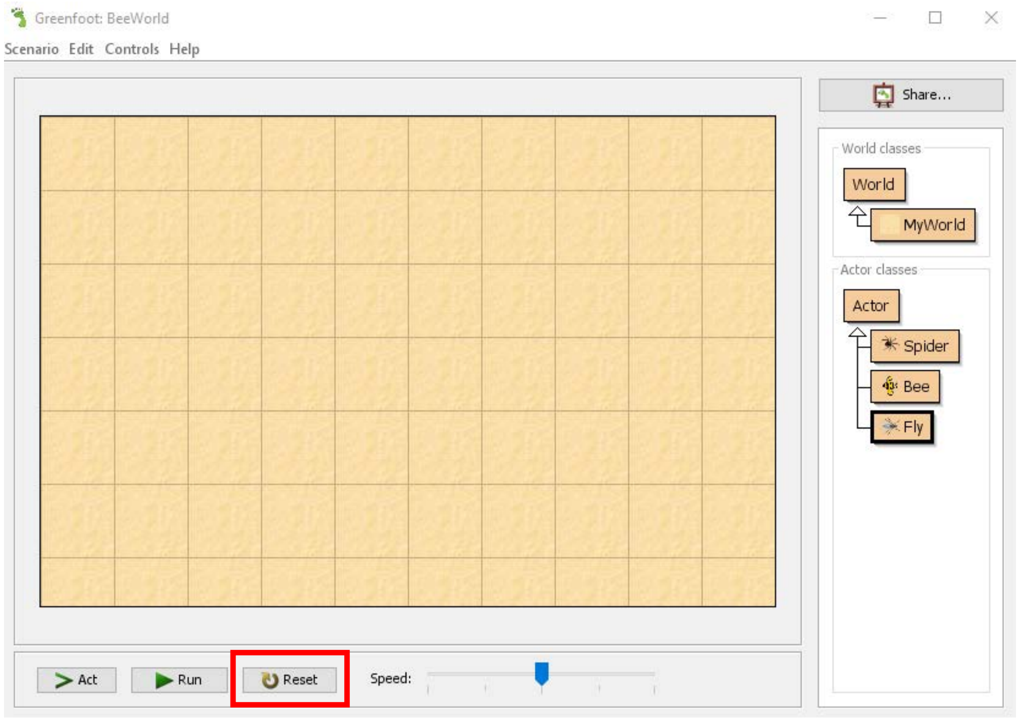
Steps to Add an Instance to a Scenario

- Right+click and add one instance of the Bee, Fly, and Spider
- Place them in your scenario so that it looks similar to this:



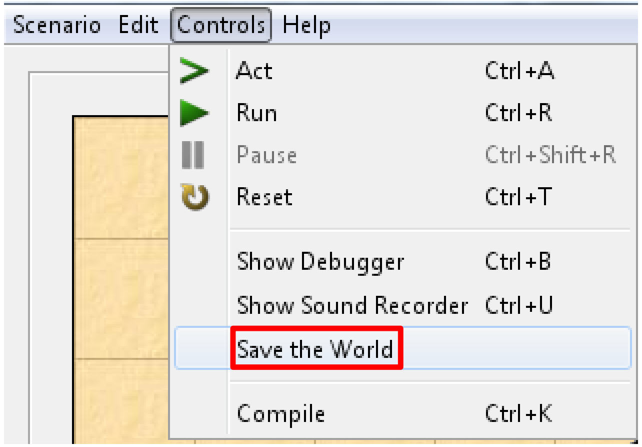
Steps to Add an Instance to a Scenario

- Now, click the “Reset” button
- What happened???



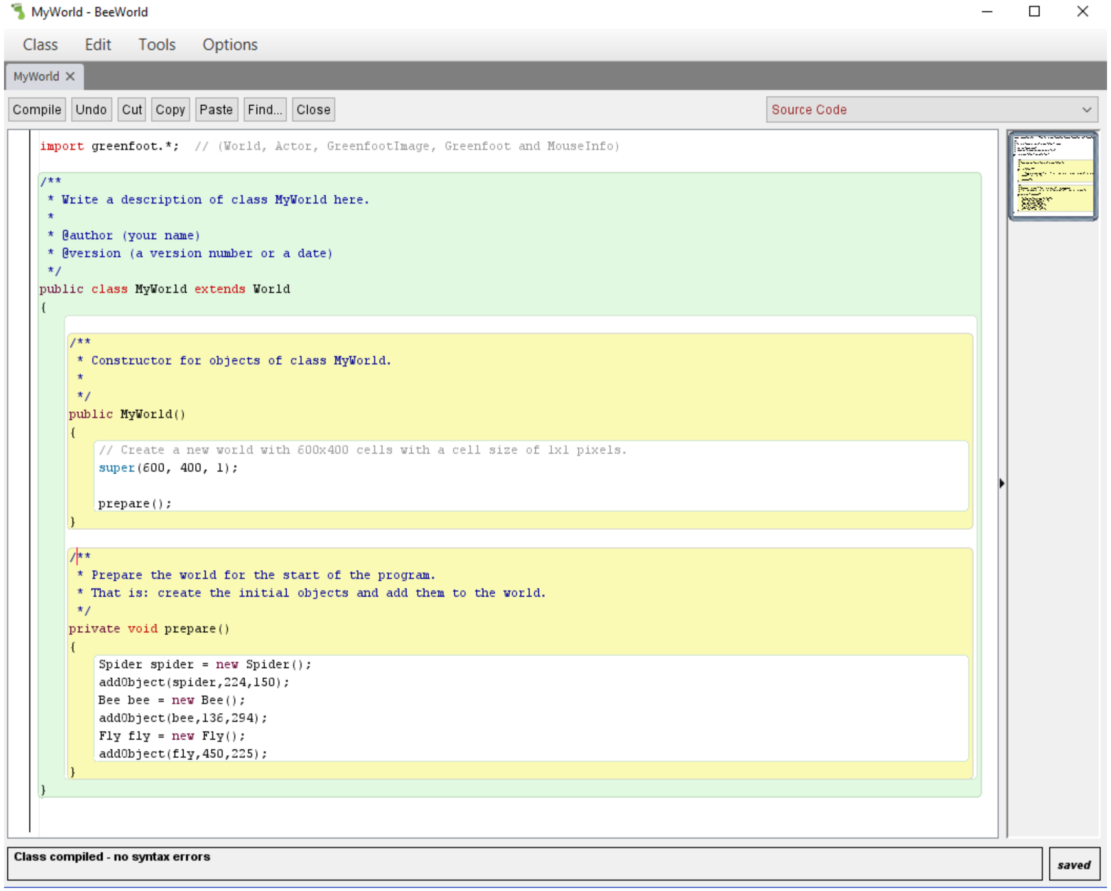
Steps to Add an Instance to a Scenario

- Adding Instances from the Scenario Actor list is only temporary!
- To add them so that they remain in the Scenario:
  - Right+click and add one instance of the Bee, Fly, and Spider
  - Click the “Controls” menu, and select “Save the World”



Steps to Add an Instance to a Scenario

- A new screen will open, showing the Java code for the World:

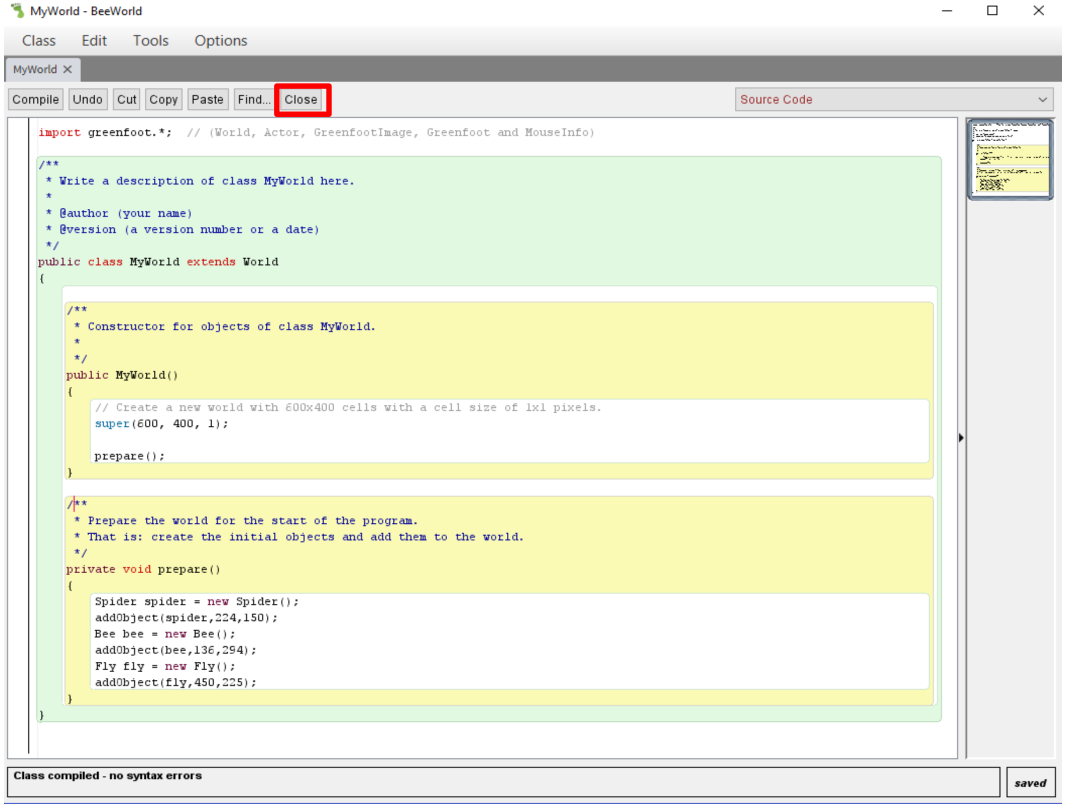


```
import greenfoot.*;  // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class MyWorld here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class MyWorld extends World
{

    /**
     * Constructor for objects of class MyWorld.
     *
     */
    public MyWorld()
    {
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(600, 400, 1);

        prepare();
    }

    /**
     * Prepare the world for the start of the program.
     * That is: create the initial objects and add them to the world.
     */
    private void prepare()
    {
        Spider spider = new Spider();
        addObject(spider,224,150);
        Bee bee = new Bee();
        addObject(bee,136,294);
        Fly fly = new Fly();
        addObject(fly,450,225);
    }
}
```

Class compiled - no syntax errors

Steps to Add an Instance to a Scenario

- Later in the course, you will learn more about this, but for now, Close the Code Editor



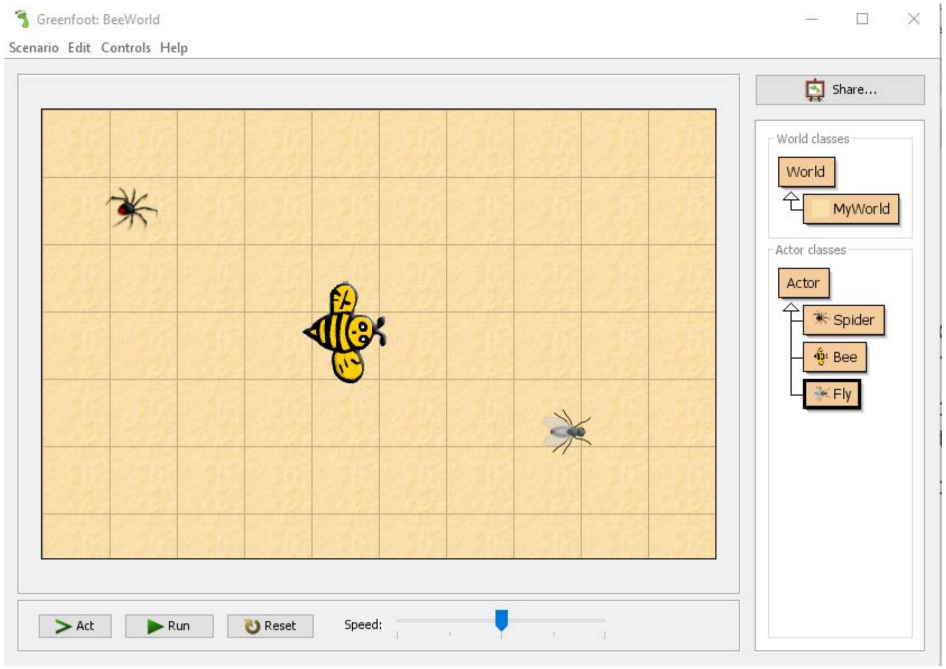
```
import greenfoot.*;  // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class MyWorld here.
 * 
 * @author (your name) 
 * @version (a version number or a date)
 */
public class MyWorld extends World
{

    /**
     * Constructor for objects of class MyWorld.
     * 
     */
    public MyWorld()
    {    
        // Create a new world with 600x400 cells with a cell size of 1x1 pixels.
        super(600, 400, 1); 

        prepare();
    }

    /**
     * Prepare the world for the start of the program.
     * That is: create the initial objects and add them to the world.
     */
    private void prepare()
    {
        Spider spider = new Spider();
        addObject(spider,224,150);
        Bee bee = new Bee();
        addObject(bee,136,294);
        Fly fly = new Fly();
        addObject(fly,450,225);
    }
}
```

Class compiled - no syntax errors

saved

Steps to Add an Instance to a Scenario

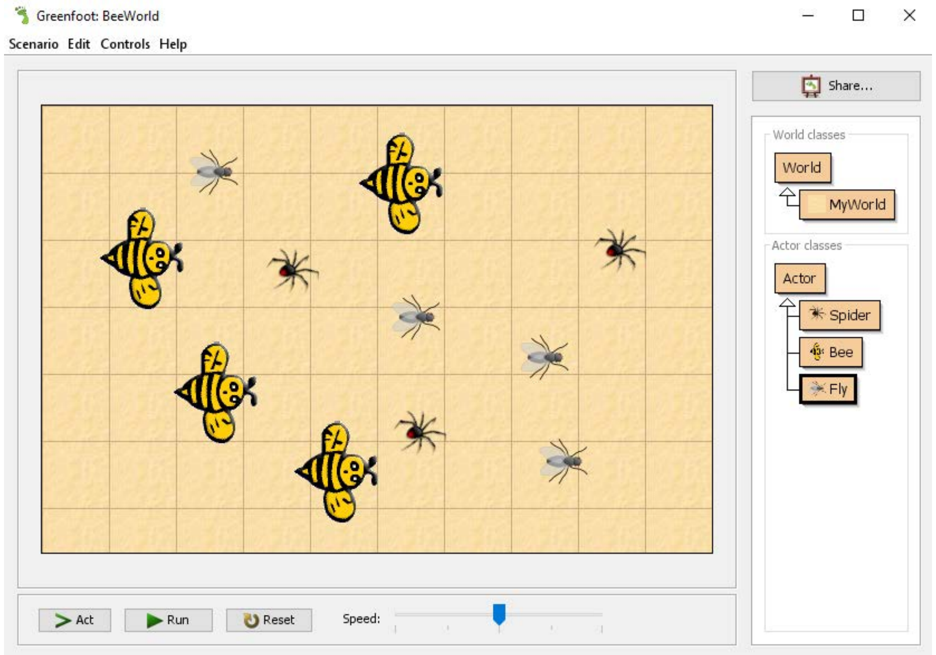
- Now, click the “Reset” button again
- What happened???



SUCCESS!!!

Steps to Add an Instance to a Scenario

- You can still move the Instances around or add new Instances
- Just click “Save the World” each time to keep the layout



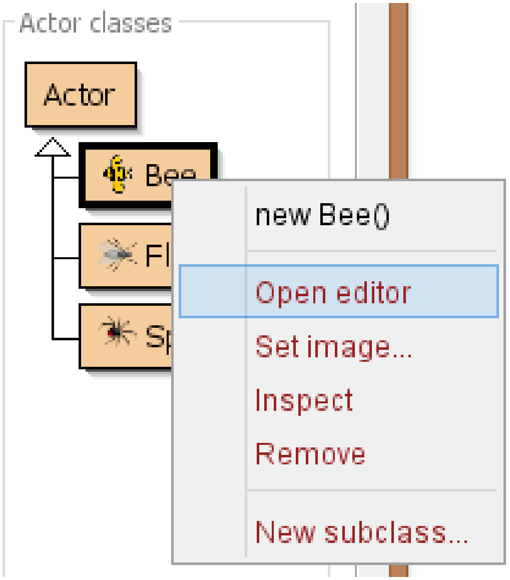
Source Code

- DNA gives humans certain characteristics, such as appearance, mobility, and communication.
- Like DNA, source code is written to tell the class how its instances could act in the scenario.

Source code defines what all instances of each class are capable of doing. The behavior of each instance is determined by the source code of its class.

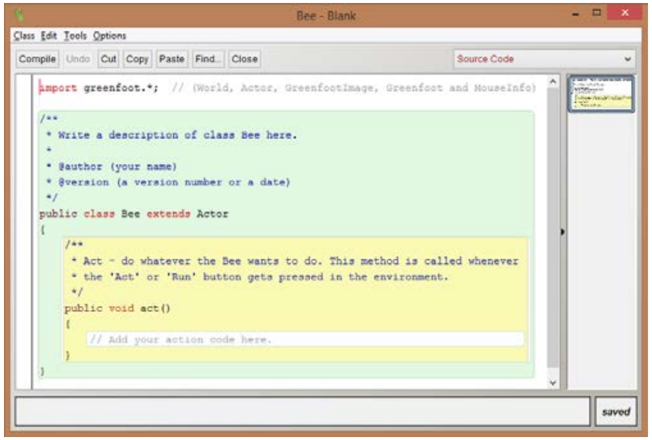
Steps to View a Class's Source Code

- Right click on a class in the class menu.
- Select Open Editor.



Code Editor

- The Code editor displays the class's source code.
- This is where instructions are programmed for how instances of the class can act in the scenario.
- In the following lessons, you will learn how to edit the Java code for your Greenfoot projects



```
import greenfoot.*;  // (World, Actor, GreenfootImage, Greenfoot and MouseInfo)

/**
 * Write a description of class Bee here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
public class Bee extends Actor
{
    /**
     * Act - do whatever the Bee wants to do. This method is called whenever
     * the 'Act' or 'Run' button gets pressed in the environment.
     */
    public void act()
    {
        // Add your action code here.
    }
}
```

Terminology

Key terms used in this lesson included:

- Class
- Compilation
- Instance
- Source code
- Subclass
- Superclass

Summary

In this lesson, you should have learned how to:

- Download and install Greenfoot
- Describe the components of the Greenfoot interactive development environment
- Create an instance of a class
- Describe classes and subclasses
- Recognize Java syntax used to correctly create a subclass

ORACLE® ACADEMY